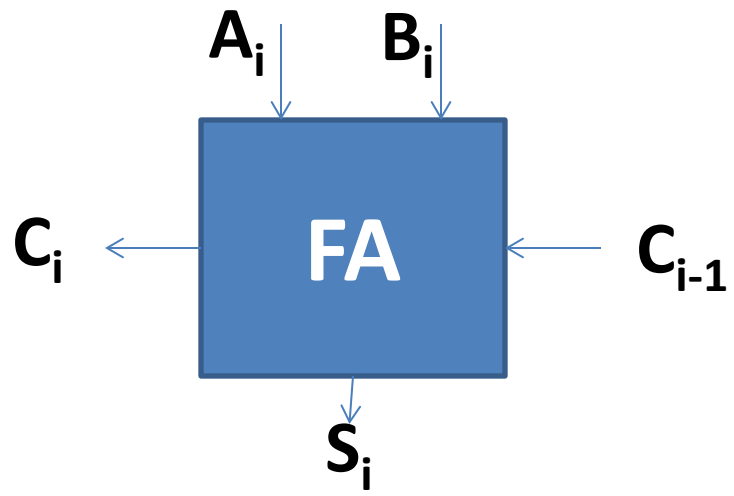
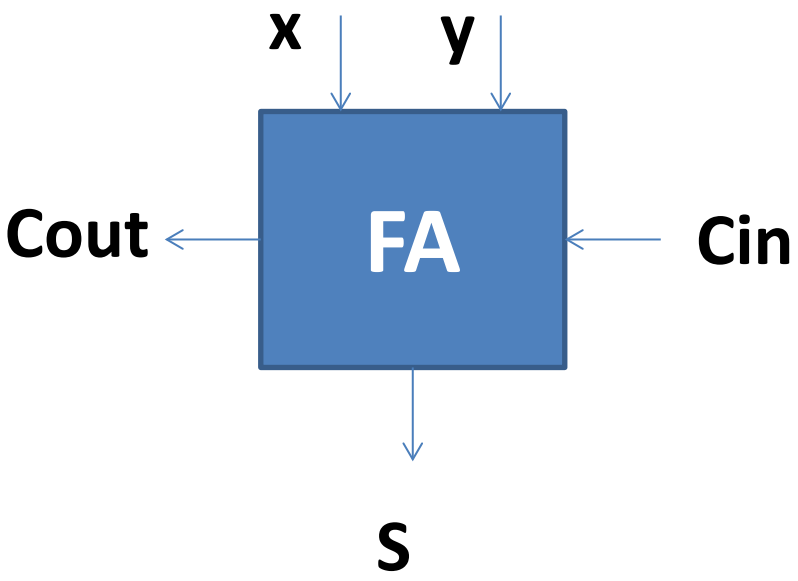
Addition

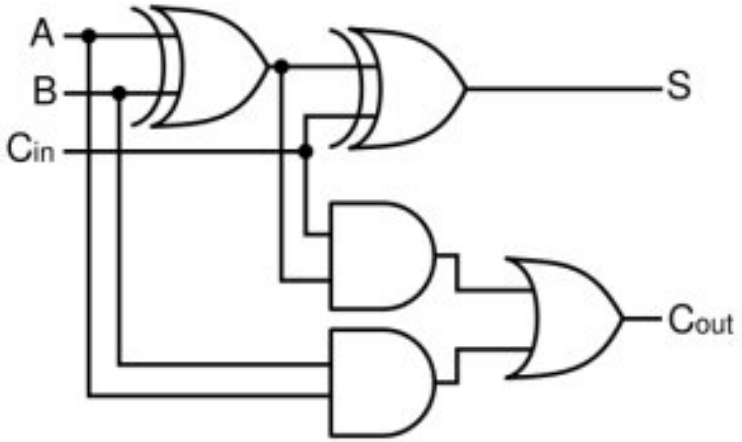
```
  1 0 1 1 0
  1 1 0 0 1
-----------
1 0 1 1 1 1
```



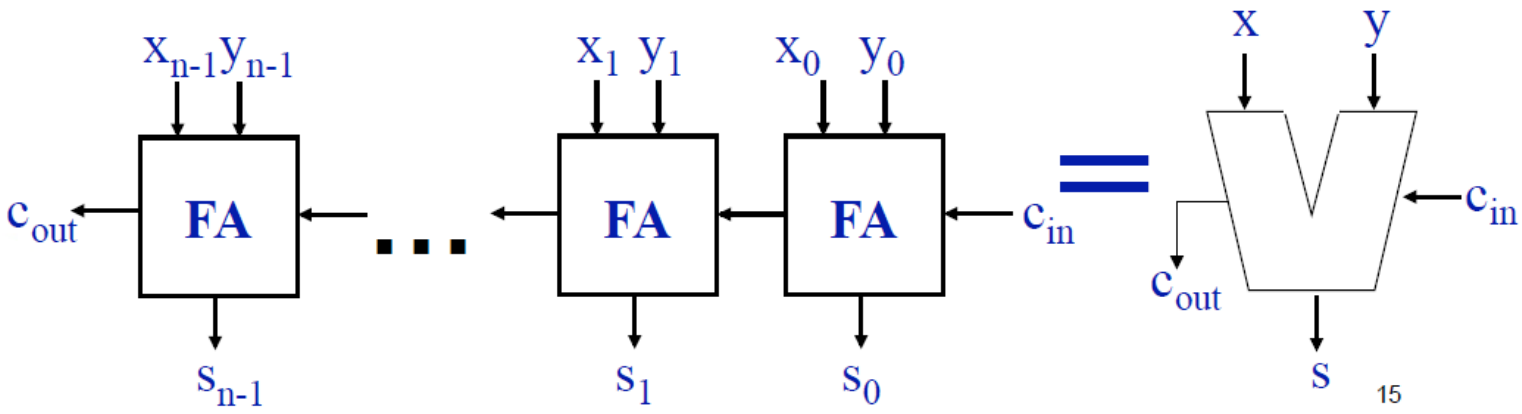
Full-Adder

- $s = A \oplus B \oplus Cin$
- $Cout = A \cdot Cin + B$



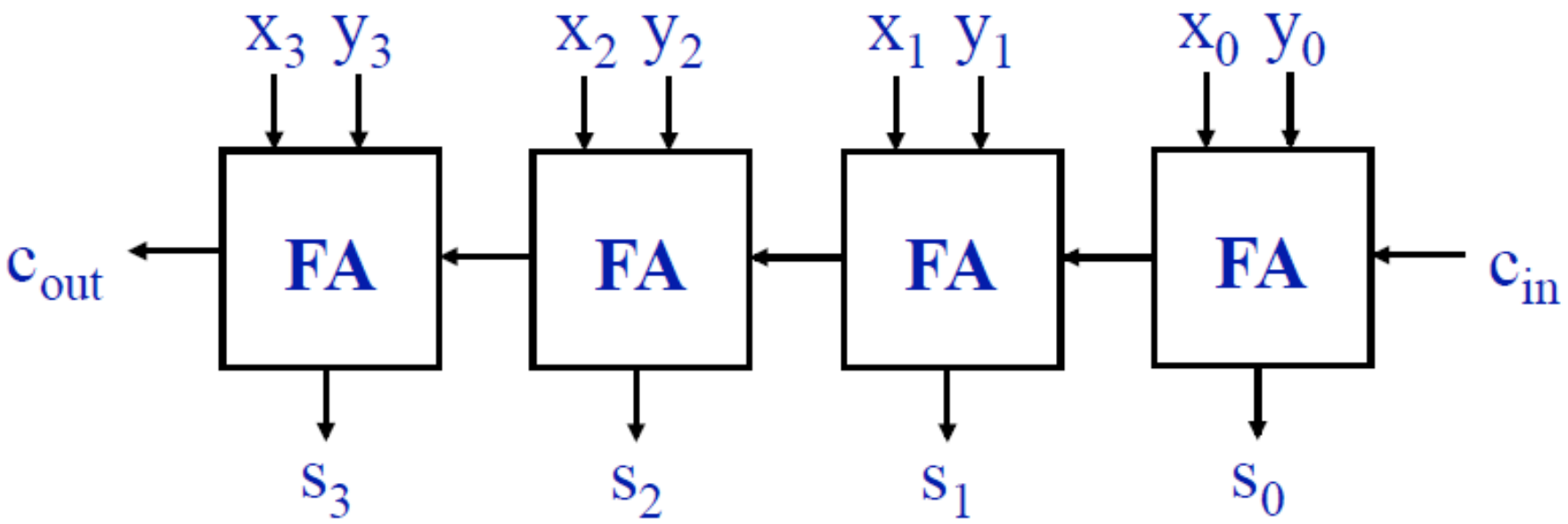


N-Bit Ripple-Carry Adder



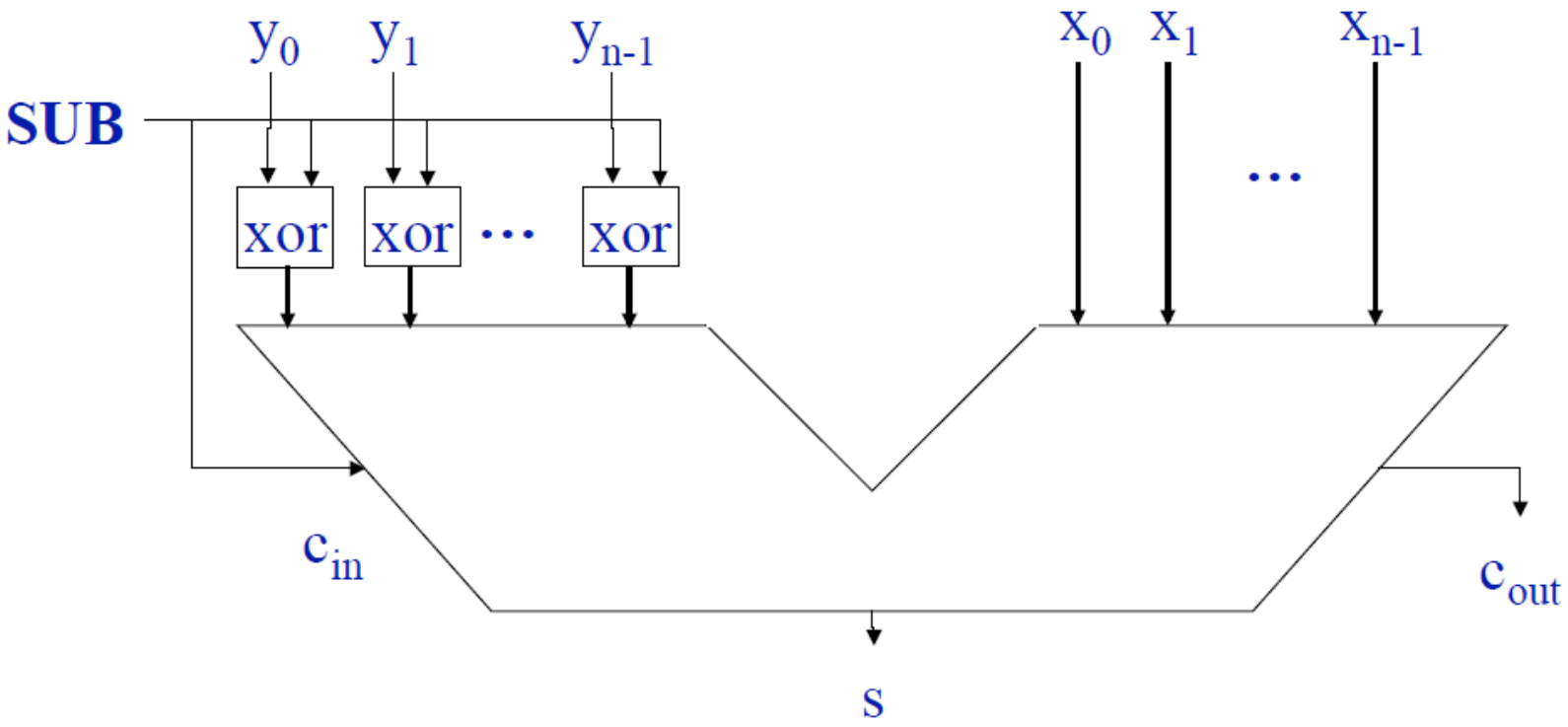
Ripple-Carry Addition

- Let's add 0111 + 1001

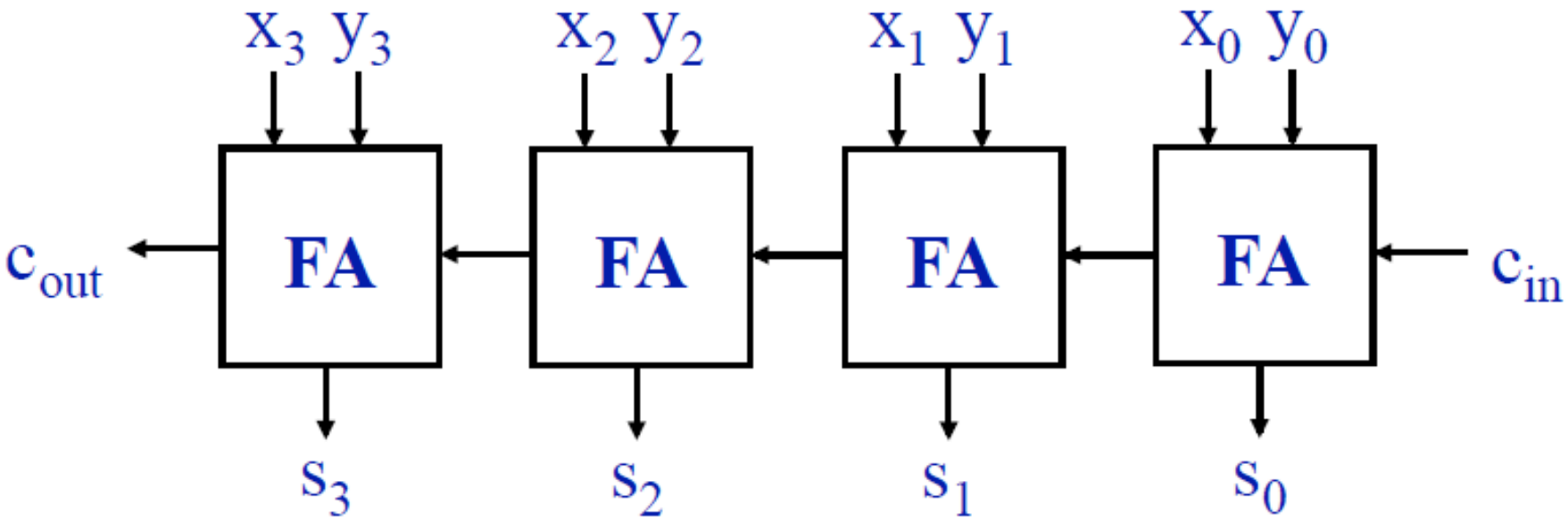


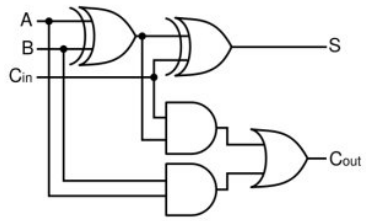
Subtraction

- Note that -x = inverse of x + 1



Delay through the Ripple-Carry Adder





- Two gates (AND and OR) Cin
- Delay ~ 2 x N
- N bits

A Faster Adder

- Each Stage either:
  - Generates a Carry : 1 + 1
  - Propagates a Carry: 1 + 0, 0 + 1, 0 + 0

$$C_{i+1} = x_i c_i + y_i c_i + x_i y_i$$

$$C_{i+1} = \underbrace{(x_i + y_i)}_{\text{Propagate}} c_i + \underbrace{x_i y_i}_{\text{Generate}}$$

Carry-Lookahead Adder

- Let:

$$G_i = x_i y_i$$

$$P_i = (x_i + y_i)$$

$$C_{i+1} = P_i c_i + G_i$$

- **P** is called the **Propagate** signal. **G** is called the **Generate** signal.
  - The carry input to each stage can be calculated as follows:

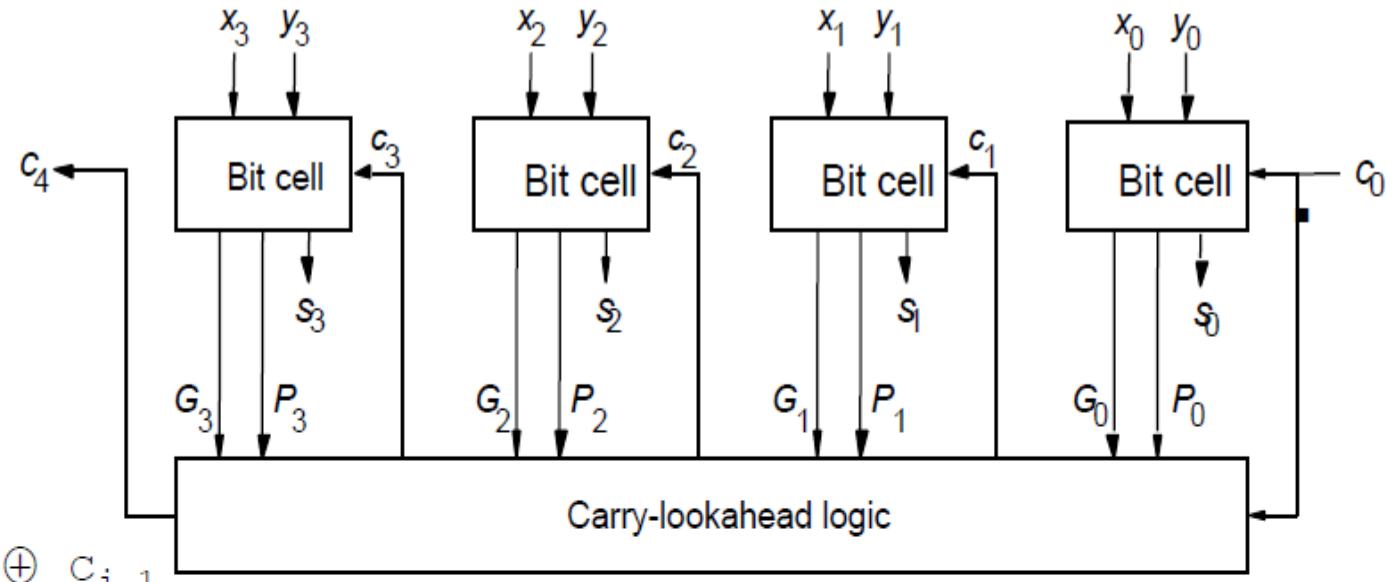
$$c_1 = P_0 c_{in} + G_0$$

$$c_2 = P_1 c_1 + G_1$$

$$= P_1 P_0 c_{in} + P_1 G_0 + G_1$$

$$c_3 = P_2 P_1 P_0 c_{in} + P_2 P_1 G_0 + P_2 G_1 + G_2$$

4-Bit CLA Adder



Bit cell:

$G_i = x_i y_i$

$P_i = (x_i + y_i)$

$s_i = x_{i-1} \oplus y_{i-1} \oplus c_{i-1}$

CLA Logic:

$$c_1 = P_0 c_{in} + G_0$$

$$c_2 = P_1 P_0 c_{in} + P_1 G_0 + G_1$$

$$c_3 = P_2 P_1 P_0 c_{in} + P_2 P_1 G_0 + P_2 G_1 + G_2$$

$$c_4 = P_3 P_2 P_1 P_0 c_{in} + P_3 P_2 P_1 G_0 + P_3 P_2 G_1 + P_3 G_2 + G_3$$

CLA Adder Delay Analysis

$$c_1 = P_0 c_{in} + G_0$$
$$c_2 = P_1 P_0 c_{in} + P_1 G_0 + G_1$$
$$c_3 = P_2 P_1 P_0 c_{in} + P_2 P_1 G_0 + P_2 G_1 + G_2$$
$$c_4 = P_3 P_2 P_1 P_0 c_{in} + P_3 P_2 P_1 G_0 + P_3 P_2 G_1 + P_3 G_2 + G_3$$

- There are a couple of key features to note with this reorganization:
  - Each P and G signal only depends on the inputs X and Y. They are available 1 gate delay after the inputs to the adder are available.
  - Looking at the expression:

    $$c_2 = P_2 P_1 P_0 c_{in} + P_2 P_1 G_0 + P_2 G_1 + G_2$$

    Lets compute the delay of the critical path

CLA Adder Delay Analysis

$$c_1 = P_0 c_{in} + G_0$$
$$c_2 = P_1 P_0 c_{in} + P_1 G_0 + G_1$$
$$c_3 = P_2 P_1 P_0 c_{in} + P_2 P_1 G_0 + P_2 G_1 + G_2$$
$$c_4 = P_3 P_2 P_1 P_0 c_{in} + P_3 P_2 P_1 G_0 + P_3 P_2 G_1 + P_3 G_2 + G_3$$

- The delay for all carry signals is 3 gate delays:
  1. 1 OR for the P terms and 1 AND for the G terms (in parallel)
  2. 1 AND to and all P's and the carry-in
  3. 1 OR compute the carry-out for the stage

- Now, the last sum term actually comes out later since it still requires an XOR after the carry is produced
  - $s_i = x_i \oplus y_i \oplus c_i$

- Final sum is available in 4 gate delays

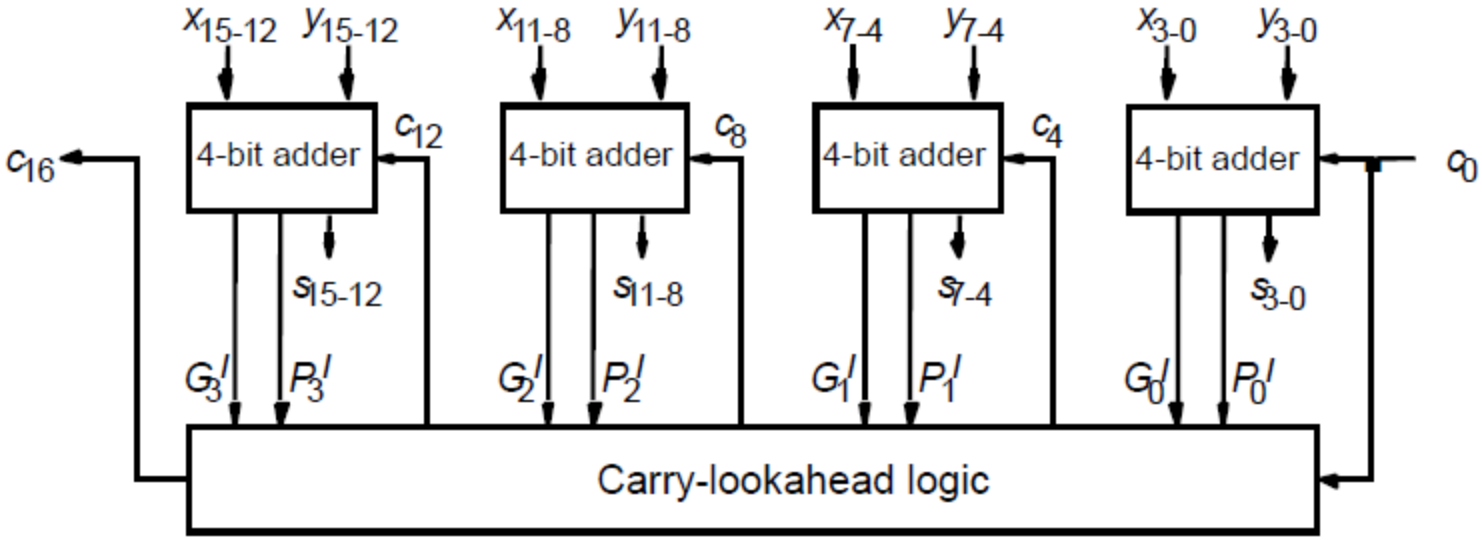
Compound CLA Adder

- Clearly the carry look-ahead adder has a speed advantage, at the cost of more gates:
  - The additional gates calculate many of the intermediate results in parallel instead of serially
  - More gates will result in more silicon area and ultimately more power.

- Another problem is that the number of inputs (fan-in) into a gate cannot be increased indefinitely:
  - Beyond a 5 input gate, the gate starts to take too long
  - **Simple gate delay approximation breaks down…**

We can combine several CLA's to keep the fan-in low

16-bit CLA Adder



Each 4-bit adder is a 4-bit CLA adder with additional outputs:

$$G_0' = P_3P_2P_1G_0 + P_3P_2G_1 + P_3G_2 + G_3$$

$$P_0' = P_3P_2P_1P_0$$

$$C_4 = P_0'c_{in} + G_0'$$

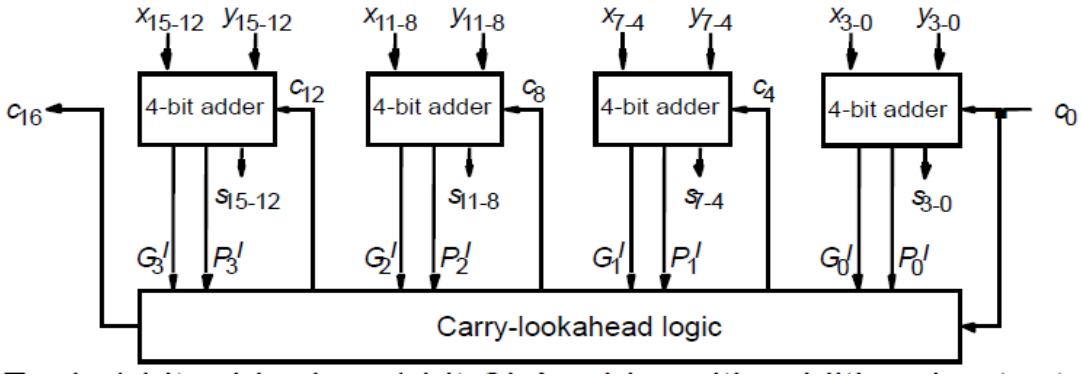
$$G_1' = P_7P_6P_5G_4 + P_7P_6G_5 + P_7G_6 + G_7$$

$$P_1' = P_7P_6P_5P_4$$

$$C_8 = P_1'P_0'c_{in} + P_1'G_0' + G_1'$$

$$= P_7P_6P_5P_4P_3P_2P_1P_0c_{in} + P_7P_6P_5P_4P_3P_2P_1G_0 + P_7P_6P_5P_4P_3P_2G_1 + \ldots + P_7G_6 + G_7$$

Delay of 16-bit CLA Adder



- Calculated the delay through this adder:
  1. All P_X and G_X terms are available 1 gate delay after X and Y are available.
  2. All P_X' & G_X' terms ready 2 gate delays after P_X & G_X terms.
  3. C_4, C_8, C_{12}, C_{16} available 2 gate delays after P_X' & G_X' terms.
  4. Internal carries (i.e. C_5, C_6, C_7) take another 2 gate delays
     - You can't calculate these until C_4, C_8, C_{12} are ready
     - i.e. $C_5 = P_4 \mathbf{c_4} + G_4$
  5. Final sum takes 1 more gate delay after C_7, C_{11}, C_{15} are done
     - $s_7 = x_7 \oplus y_7 \oplus c_7$

- In total we have delay = 1 + 2 + 2 + 2 + 1 = 8 gate delays

Carry Select Adder

- Say you wanted to calculate a 64-bit add
- But you cannot fit it in a cycle
- But you can fit 32-bit adds
- What can you do?

Carry Select Adder

- Calculate three sums in parallel
- Lower 32-bit sum
  - This produces a carry out at the 32-bit position
- Two Upper 32-bit sums
  - One that uses a 0 carry in
  - Another that uses a 1 carry in
- Once the Lower completes
  - Select which upper result is right